12-1995

Adaptive Methods for Distributed Video Presentation

Crispin Cowan

Shanwei Cen

Jonathan Walpole
Portland State University

Carlton Pu

Adaptive Methods for Distributed Video Presentation*

Crispin Cowan, Shanwei Cen, Jonathan Walpole, and Calton Pu
Department of Computer Science and Engineering
Oregon Graduate Institute of Science and Technology
Portland, Oregon, USA
{*crispin, scen, walpole, calton*}*@cse.ogi.edu*

This paper describes problems and solutions for delivering real-time, multi-media presentations across the Internet. A key characteristic of presentations of continuous media datatypes, such as digital video and audio, is their need for predictable real-time data delivery. For example, an NTSC quality video presentation requires video frames to be displayed every 1/30th of a second. Variations in this display rate can be observable as stalls or glitches in the video stream and reduce the quality of the presentation [6].

Delivering such presentations across the Internet is difficult because highly variable bandwidth and latency make it difficult to predict the arrival time of network packets containing video or audio data. Our solution is for distributed multi-media systems to *adapt* dynamically to these changing network conditions. This paper describes the use of *software feedback* to make multimedia presentations adaptive, and shows how video can be played across an unpredictable network such as the Internet without benefit of resource reservations.

The Internet's unpredictable latency and bandwidth characteristics arise because different links in the network have performance that varies by several orders of magnitude. Hence, the location of a video client relative to its video server influences the performance characteristics of the connection. Furthermore, even if the capability of the hardware in question can be established, the *available* bandwidth varies wildly from moment to moment because the Internet is a shared resource: just a few concurrent large data transfers can easily take up most of a connection's bandwidth. In this environment, adaptive methods are essential to maintaining video quality.

*This project is supported in part by grants from ARPA and the National Science Foundation, and donations from Tektronix, Hewlett-Packard and the Portland Trail Blazers.

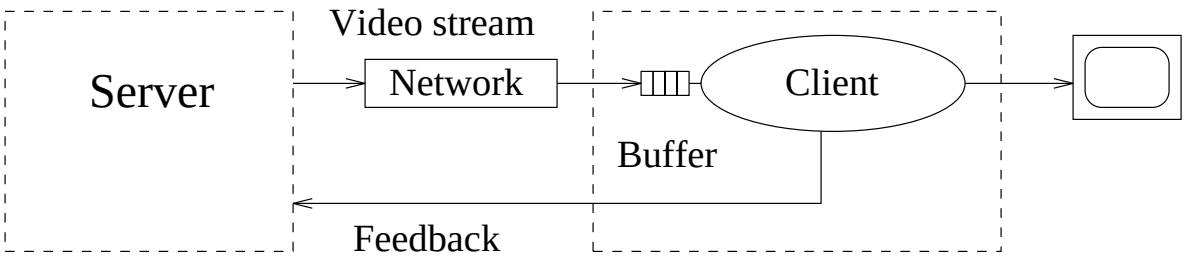

Figure 1: Architecture of the player

We use **software feedback** [3, 4], reminiscent of hardware feedback, to *adapt* multimedia presentations to the changing conditions of the Internet [1]. Our video player has a distributed client-server architecture as shown in Figure 1. The client measures various properties of the video stream it is receiving from the network, and feeds them back to the server, allowing both the client and the server to adapt to changing Internet conditions. This design allows videos to be played, in real-time, across the Internet without having to first down-load the video to a local file.

Software feedback has been employed in many forms, such as the flow control mechanisms in TCP, the clock synchronization mechanism used in NTP, and Rowe's video stream frame rate control mechanism [5]. However, these implementations have all used feedback as an *ad hoc* solution. Consequently they exhibit arbitrary structure, hard-to-predict behavior, and wasted effort due to repeated design and implementation of logically similar components. Feedback systems are hard to build correctly, and building several of them is harder.

We are developing a **software feedback toolkit** [4] to address these problems. The toolkit makes feedback systems more predictable, easier to develop, manage and tune, and reduces duplication of effort. It consists of a number of software modules that can be plugged in as needed, rather than integrating feedback directly into the multi-media components. The modules include assorted filters, comparators, and action modules useful in constructing a feedback system. For example, Figure 2 shows how basic components can be assembled to manage jitter in a video presentation.

The client continuously measures the relevant characteristics of the data it is receiving from the server, and reports these characteristics to the server, as shown in Figure 1. The Server, in turn, uses the feedback information to adapt the data stream it is sending to the client to match recent network conditions. Feedback allows the system to adapt to current network conditions by adjusting the quality of service in several different dimensions using

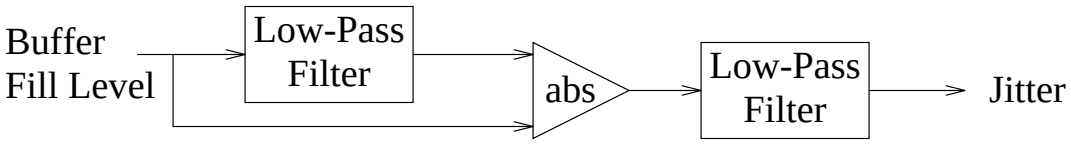

Figure 2: Composing Feedback Tools to Measure Jitter

components of the feedback toolkit:

Frame Rate For real-time video, the logical presentation rate for the stream must remain constant. However, the number of frames that can be transmitted from the server to the client during some time period varies according to the available network bandwidth. Hence, the server must decide dynamically which frames to omit from the presentation and must transmit the remaining frames at a rate supported by the currently available network bandwidth. To damp oscillations in the frame rate, a low-pass filter samples the client's received frame rate before feeding the rate information back to the server.

Jitter Variations in the inter-frame display times of a video presentation can also degrade quality even when the overall frame rate is adequate. These variations are introduced by packetization (splitting or combining video frame data into network packets) and by network congestion, making the network bursty, and also causing the degree of burstiness to vary dynamically. To prevent jitter from appearing in the video presentation, the client dynamically adjusts its buffer space to be sufficient to mask the current degree of burstiness. Figure 2 shows how the buffer fill level is converted to a jitter measure, which is used to adjust the buffer size.

Client:Server Synchronization In adjusting the frame transfer rate, the system must be careful to maintain an optimal fill level in the client's frame buffer (half-full). Hence, the client and server must be synchronized. However, the system must distinguish network jitter from per-

sistent changes in the relative positions in the video between the client and the server. Large changes in network latency can be caused by routing changes in packet-switched networks such as the Internet, and physical clock drift between the machines can also cause a loss of synchronization. Specialized filters (such as derivative filters) test the fill-level of the client's frame buffer to identify these phenomena, and distinguish them from short-term network jitter. Feedback is then used to take corrective action.

The major benefits of software feedback are improved presentation quality, and improved resource utilization. In [1] we validate the benefits of feedback to multimedia quality of service with quantitative analysis of the benefits of feedback on the playback of an MPEG movie across the Internet. In [2] we describe a demo that qualitatively demonstrates the benefits of feedback. Quality is improved in the following ways:

- Not all frames in an MPEG data stream are equally important: dropping an I-frame not only results in the loss of that frame, it also results in the loss of numerous other B- or P-frames that are derived from it. Hence, intelligently dropping B- and P-frames at the server instead of letting a congested network drop random frames improves presentation quality.

- Optimizing the buffer space used to minimize jitter improves both the "smoothness" of the presentation and reduces end-to-end latency.

- Clock drift does not affect quality until it causes complete under- or over-flow of the client's buffer, at which point the result is catastrophic [1]. Incorporating a clock synchronization mechanism into the feedback system avoids this pathology.

Resource utilization is improved in the following ways:

- By only sending frames at the rate of the weakest link in the connection, the server saves network bandwidth on all links between the server and the weak link for frames that would have been dropped. It also saves bandwidth over the entire connection for the frames that would have arrived too late to be displayed. Sending only frames that are likely to be used also conserves compute power at the server.

- Adjusting the size of the client's work-ahead buffer to be just sufficient to minimize jitter conserves real memory in the client.

A large-scale distributed environment is necessarily a shared environment, and thus presents performance and load characteristics that are highly dynamic. The real-time requirements of multi-media applications demand that they have some ability to adapt to these changing load conditions. Our work has shown how software feedback can benefit the performance of a distributed video player, and how the systematic packaging of software feedback technology can aid in applying adaptive methods more easily and effectively.

In our current and future research, we are investigating tools to allow for the automatic composition of feedback components while maintaining system stability. We are expanding the number of dimensions of quality of service that can be adjusted to manage bandwidth consumption to include resolution and dynamic rate shaping. We are also investigating the applicability of feedback-based adaptation to more complex distributed systems, such as multi-cast to multiple clients, and synchronizing data from multiple servers.

References

[1] Shanwei Cen, Calton Pu, Richard Staehli, Crispin Cowan, and Jonathan Walpole. A Distributed Real-Time MPEG Video Audio Player. In *Proceedings of the 1995 International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'95)*, pages 151–162, New Hampshire, April 1995.

[2] Shanwei Cen, Calton Pu, Richard Staehli, Crispin Cowan, and Jonathan Walpole. Demostrating the Effect of Software Feedback on a Distributed Real-Time MPEG Video Audio Player. In *Demonstration at the 1995 ACM Multimedia Conference*, pages 239–240, San Francisco, CA, November 5-9 1995.

[3] Henry Massalin and Calton Pu. Fine-Grain Adaptive Scheduling Using Feedback. *Computing Systems*, 3(1):139–173, Winter 1990.

[4] Calton Pu and Robert M. Fuhrer. Feedback-Based Scheduling: a Toolbox Approach. In *Proceedings of Fourth Workshop on Workstation Operating Systems*, Napa Valley, CA, October 1993.

[5] L.A. Rowe, K. Patel, B.C. Smith, and K Liu. MPEG Video in Software: Representation, Transmission and Playback. In *Proceedings of the International Symposium on Elec. Imaging: Science and Technology*, San Jose, CA, February 1994.

[6] Richard Staehli, Jonathan Walpole, and David Maier. Quality of Service Specifications for Multimedia Presentations. *Multimedia Systems*, 3(5/6):251–263, November 1995.